The Darwin Kernel

Session 107

The Darwin Kernel

Jim Magee
Core OS Kernel Team

Introduction

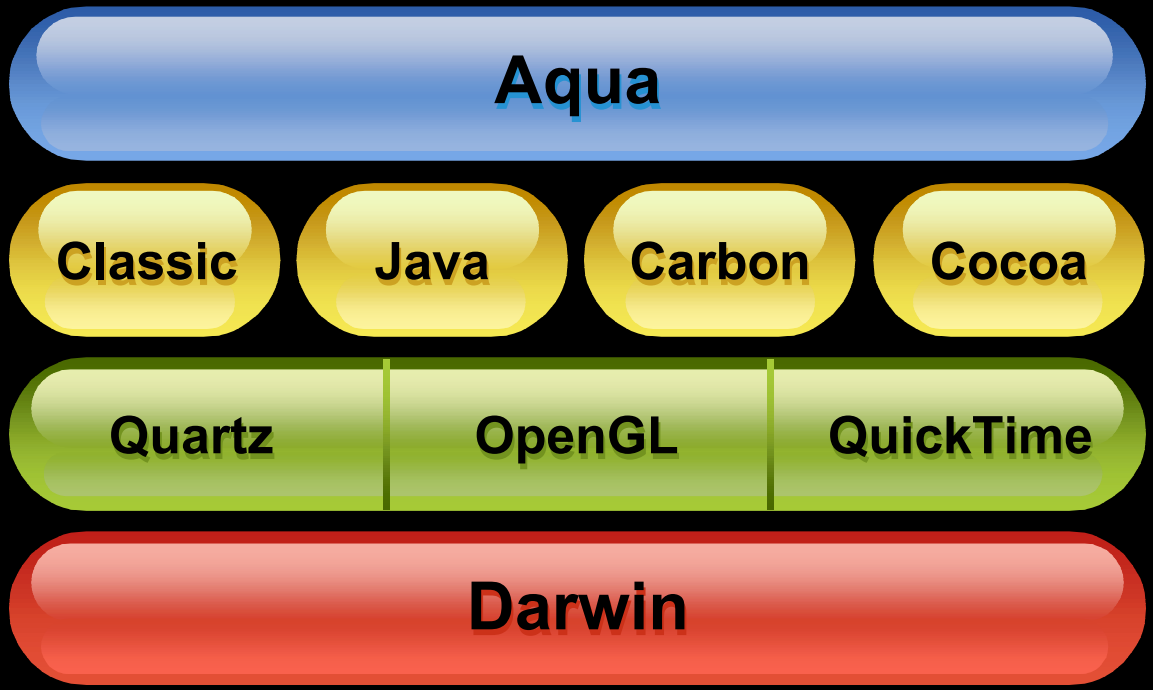
- The Darwin Kernel defines the core services in Mac OS X
- These services affect your code
    - Even when you don't directly code to them
- They are under constant refinement
    - Most often implementation
    - Sometimes semantics and syntax

What You Will Learn

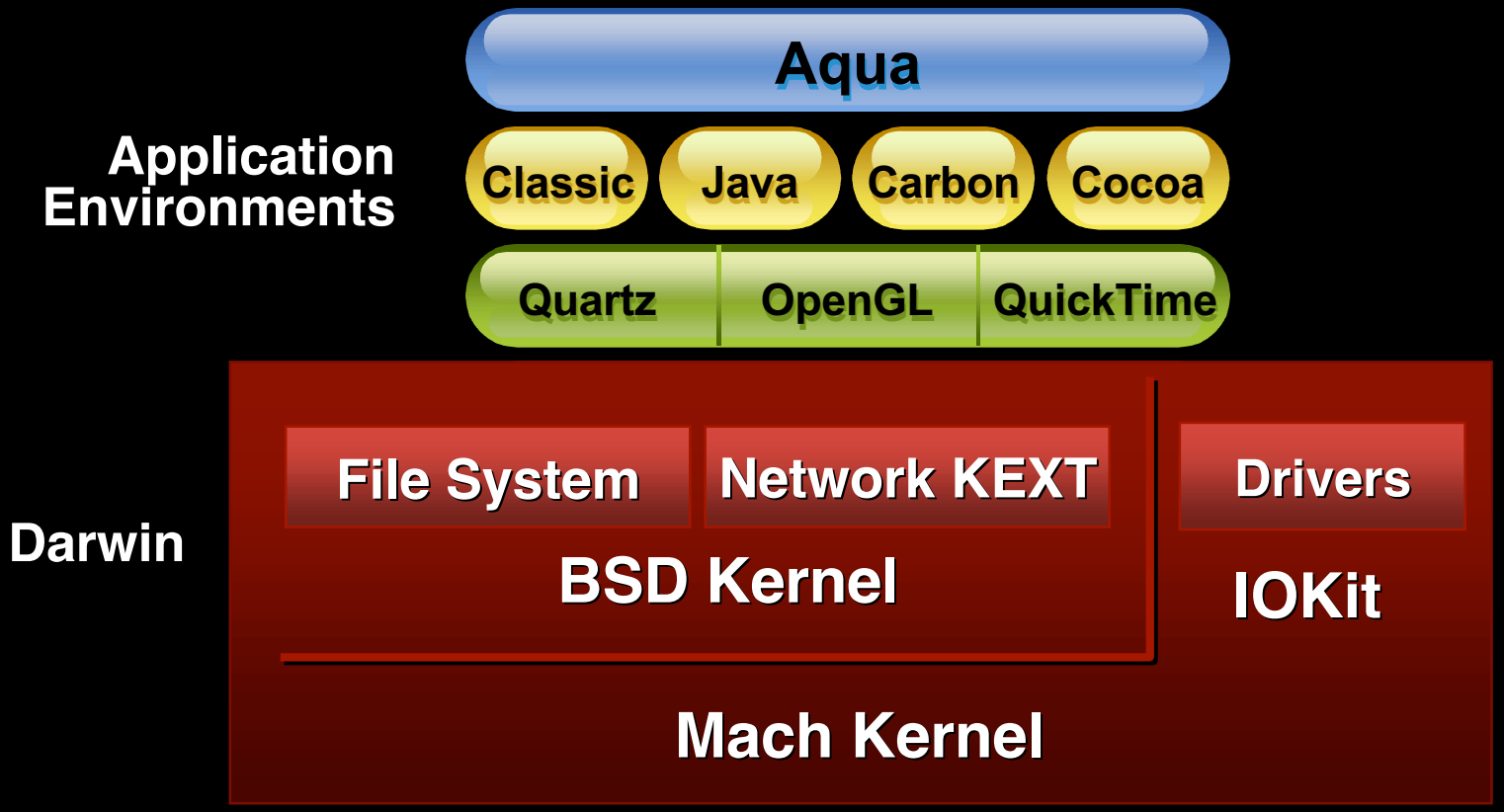
- Which services may be affecting you
    - Through a brief review of how the higher-level services layer on top
- Changes planned for Jaguar
- Future directions to consider
    - And possibly help drive

Mac OS X

Darwin technology is at the core



Darwin Kernel



Darwin Kernel

The origins

- Mach 3.0 (OSFMK73)
  - Virtual Memory, Scheduling, Inter-Process-Communication (IPC)
- 4.4BSD Lites 2
  - Process model, security, file access
- FreeBSD 3.0/3.1/3.2
  - NFS, Networking

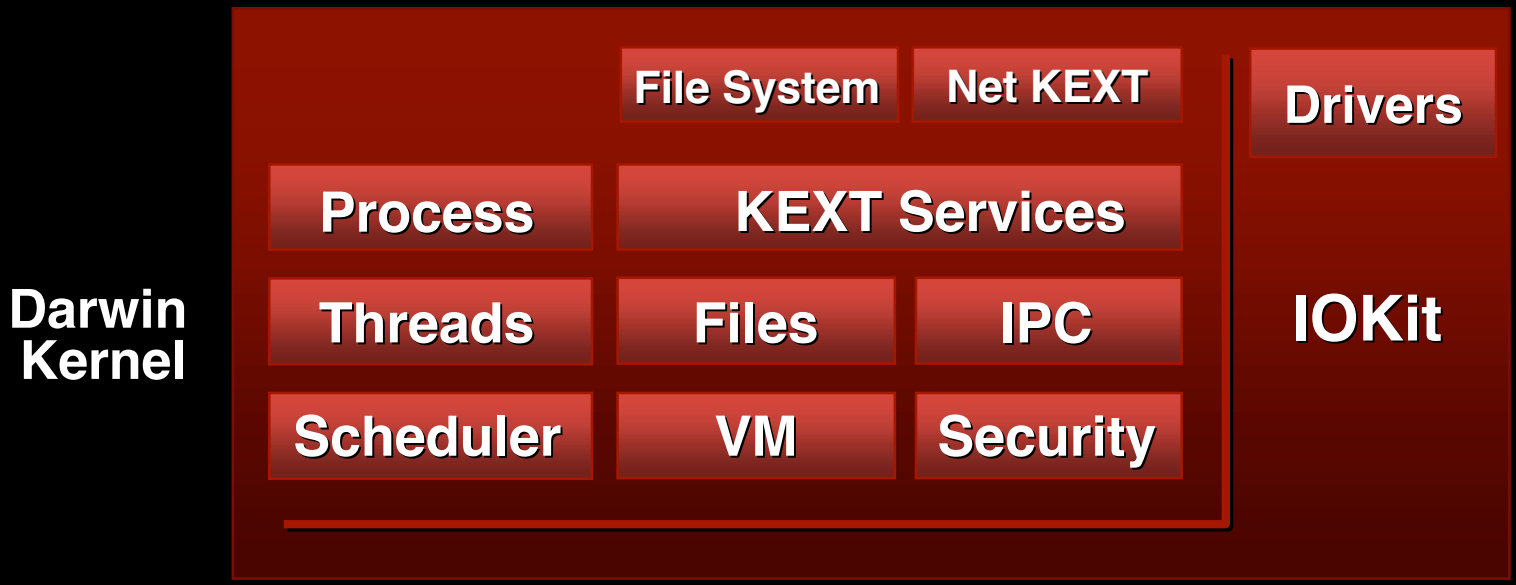
Darwin Kernel

Future directions

- Mach
  - Adopt advanced real-time scheduling
  - Drive our own changes
- BSD
  - Refresh more of kernel to FreeBSD4.4
  - Finer-grained locking

Darwin Kernel Services

Emphasis on the services, not the layering



Darwin Kernel Services

What's new for Jaguar…

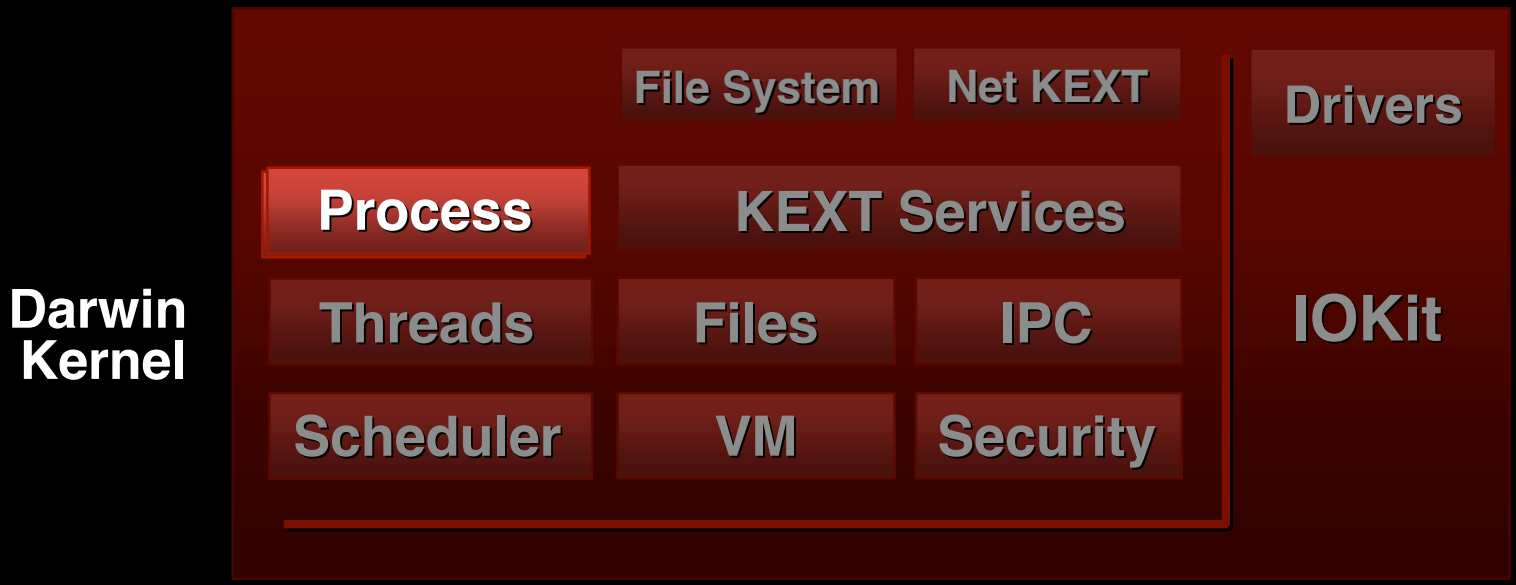
- Performance Enhancements
  - Implementation refinements
  - Better service matching
- Better Standards Adherence
  - E.g., POSIX threads
  - Porting aids (e.g., SysV IPC)
- Other New Features and Bug Fixes
  - More than 600 since last year

Darwin Kernel Services

Future Directions

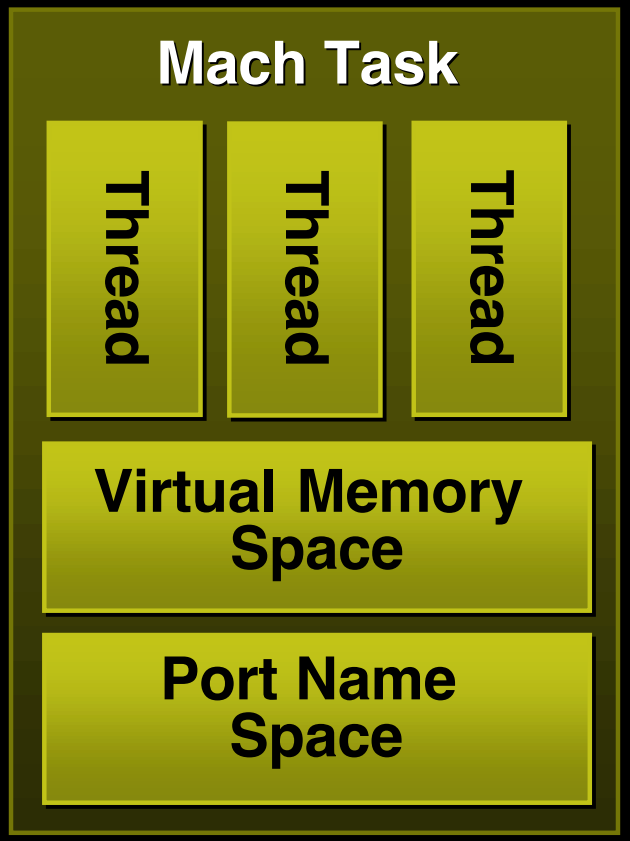
- More Performance Enhancements
  - Always
- Better Standards Adherence
  - POSIX
  - Single Unix Specification
- Rigorously-defined KEXT Services
  - Future flexibility depends on it
- Other New Features and Bug Fixes

Process Services



Mach Task Services

A task is the unit of resource ownership in Mach



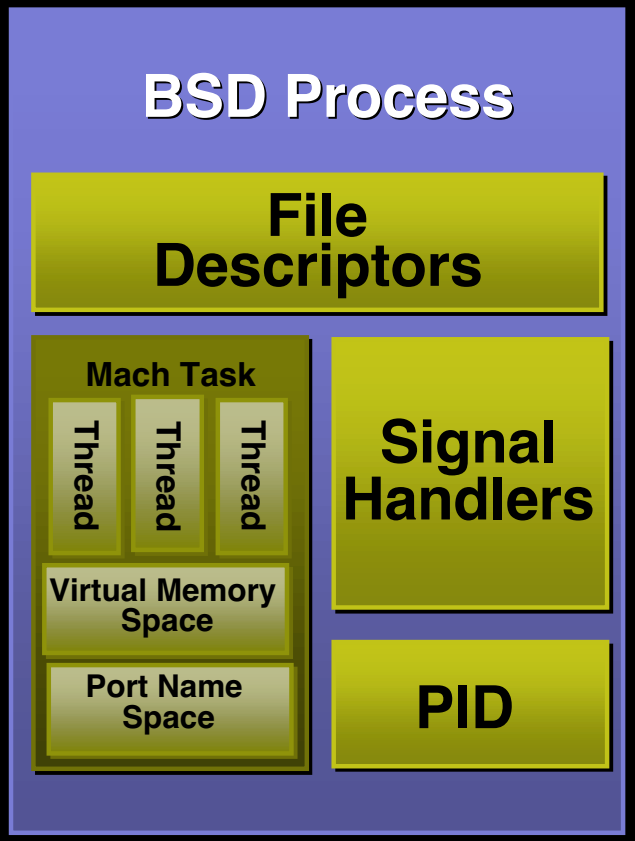
- Provide an environment in which threads run:
  - Address space
  - Communications
  - Exception handling

task_t

task_create(), task_terminate()
thread_create()
task_suspend(), task_resume()
task_swap_exception_ports()

BSD Process Services

Each BSD process contains a Mach task



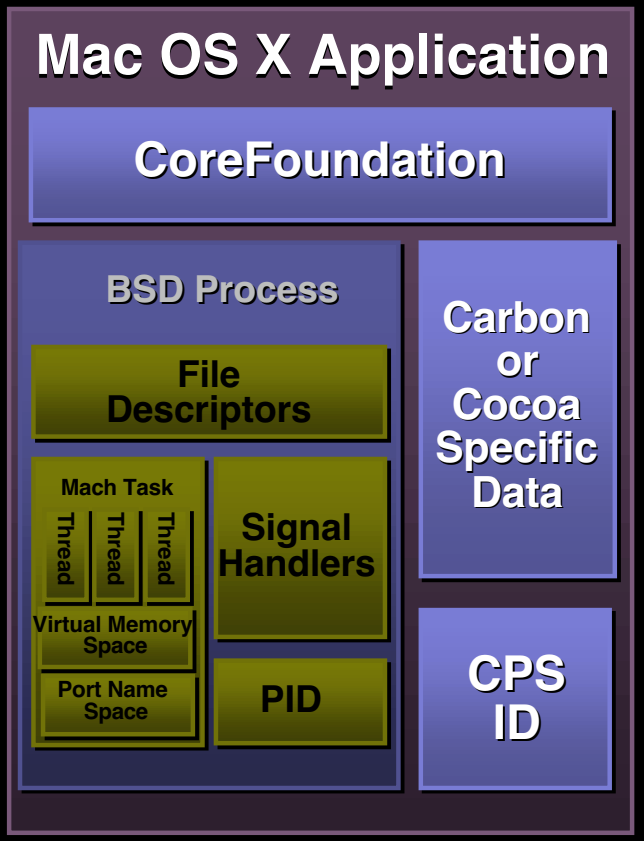
- BSD Adds:
  - File descriptors
  - Signal handling
  - Process ID
  - Process group relationships

PID

fork(), vfork(), exec(), exit()
kill(), signal()
ptrace()
task_for_pid()

Application Services

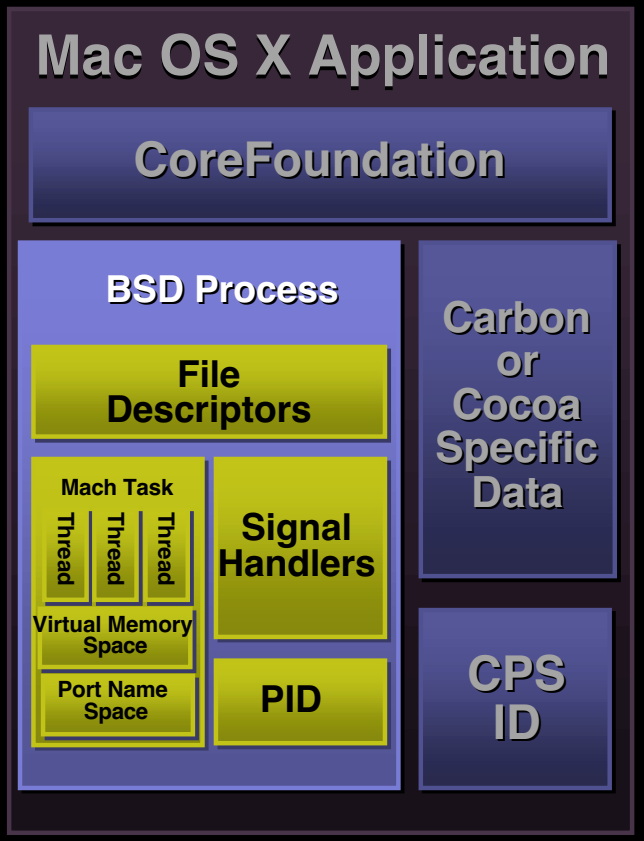
Core Services in addition to kernel facilities



- Adds:
  - Notion of "session"
  - CoreProcess management facilities
  - Launch services
- Interaction Issues:
  - Many services accessed with Mach ports
  - **fork()** without exec unsafe

Process Services

What's new from the kernel in Jaguar…

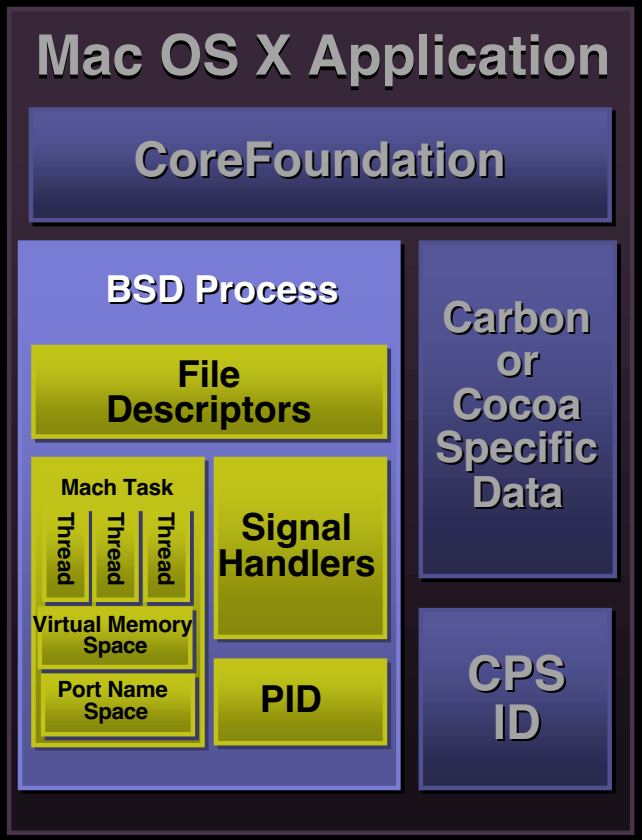


- Performance Boost
  - Process creation
  - Process exit
- POSIX Signal Support

  SA_SIGINFO, **SA_RESETHAND**, **SA_NODEFER** and **SA_NOCLDWAIT** options to **sigaction()**

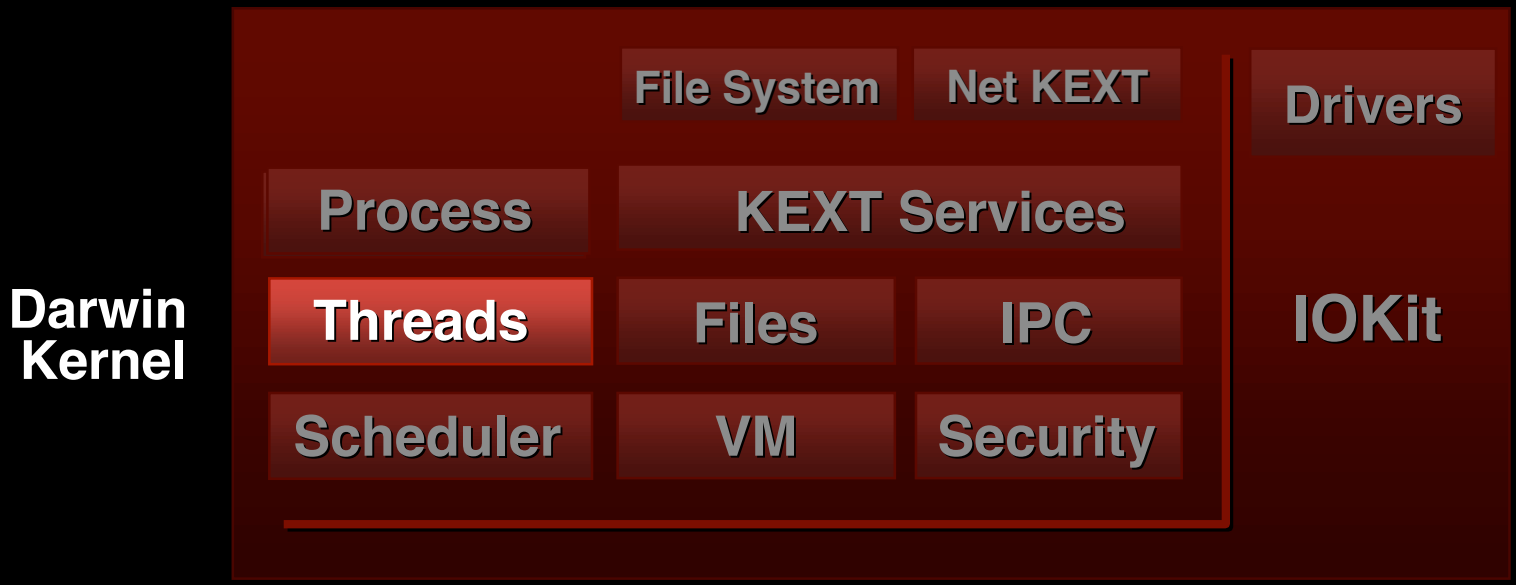
Process Services

Future Directions



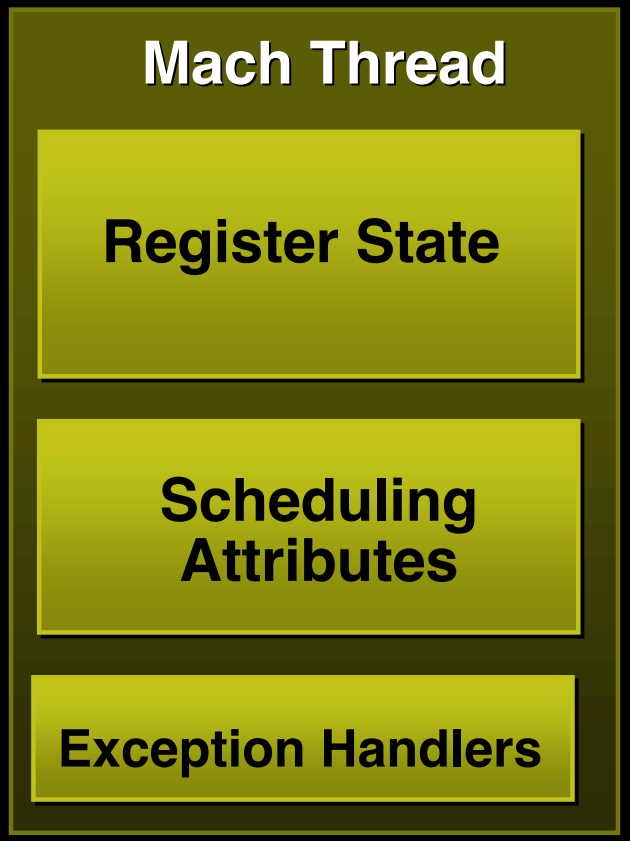
- Performance Boost/ Better Semantic Match
  - E.g., **posix_spawn()**
- More POSIX Signal Support
  - 128 signal levels
  - But no plans for real-time signal delivery at this time

Thread Services



Mach Thread Services

A thread is the unit of execution in Mach



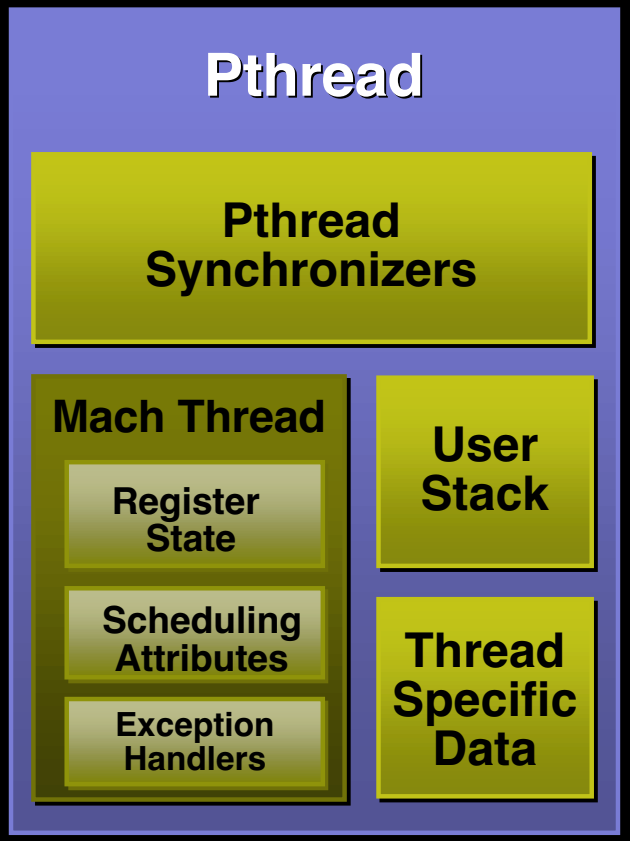
- A Mach thread owns no resources
- Defines the “how” and “where” of execution, but not the resources to execute with
- Defines thread-specific exception handling

thread_t

thread_create(), thread_terminate()
thread_suspend(), thread_resume()
thread_swap_exception_ports()

Pthread Services

POSIX Threads—The Portable Threads Layer



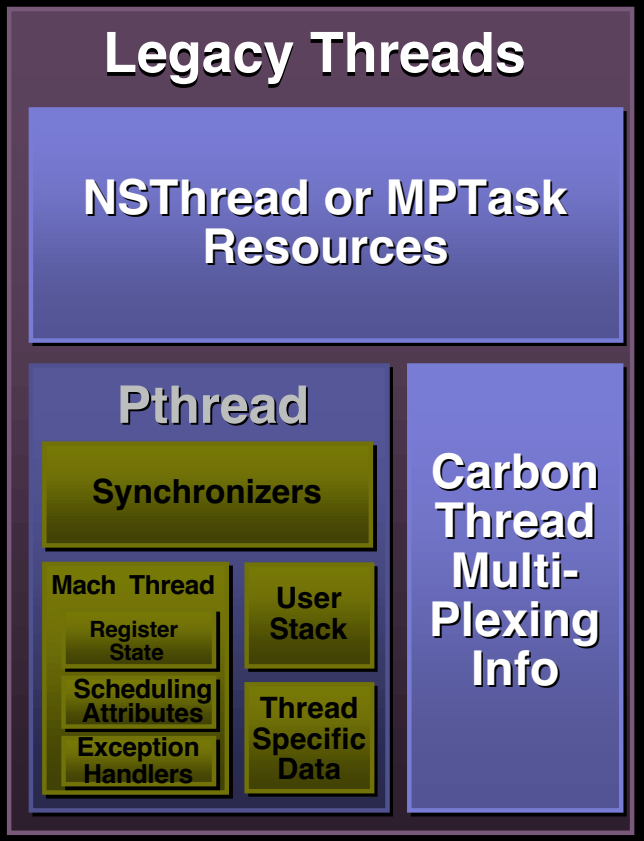
- Pthreads add synchronization
  - Mutexes
  - Conditions
- Define thread-specific resources
  - Thread stack
  - Per-thread data

pthread_t

pthread_create(), pthread_exit()
pthread_detach(), pthread_join()
pthread_cancel()

Application Threads

Provide legacy semantics



- Each MPTask or Cocoa Thread has a unique pthread
- Carbon's deferred and I/O thread environments multiplex over a handful of pthreads
  - Restricts what can be done in those environments

Thread Services

What's new from the kernel in Jaguar…

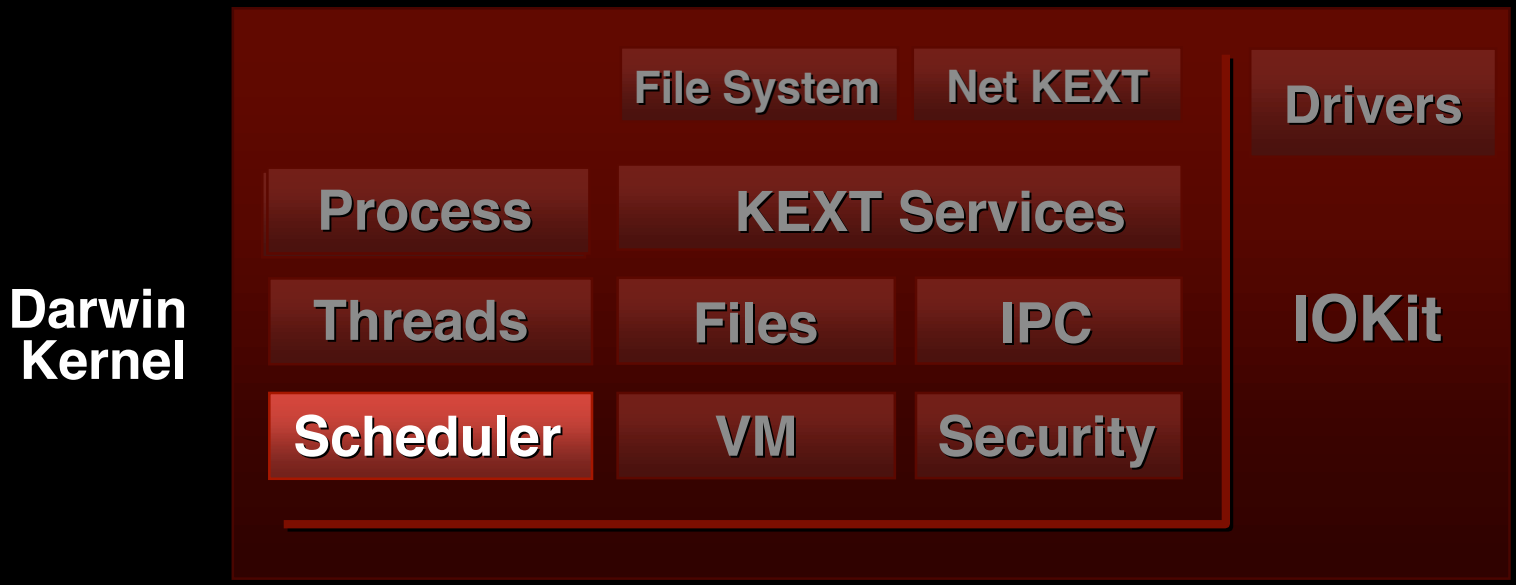
- Performance
  - Faster thread creation and termination
- Pthread synchronizers
  - Read/write locks added
  - Recursive mutexes
  - Improved pthread cancel
- Per-thread signals

Thread Services

Future Directions

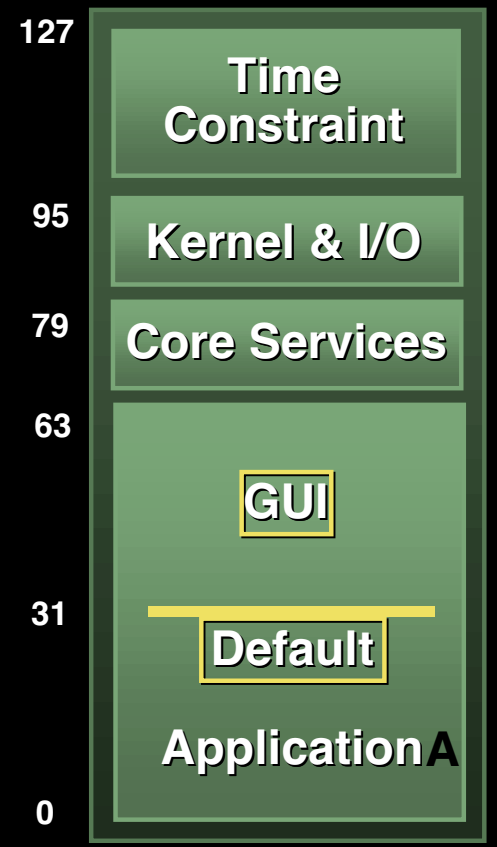
- Overall improved Pthreads standards adherence
- Mach thread name ports
  - Subset of current thread capability
  - Security driven

Scheduler Services



Scheduler Basics

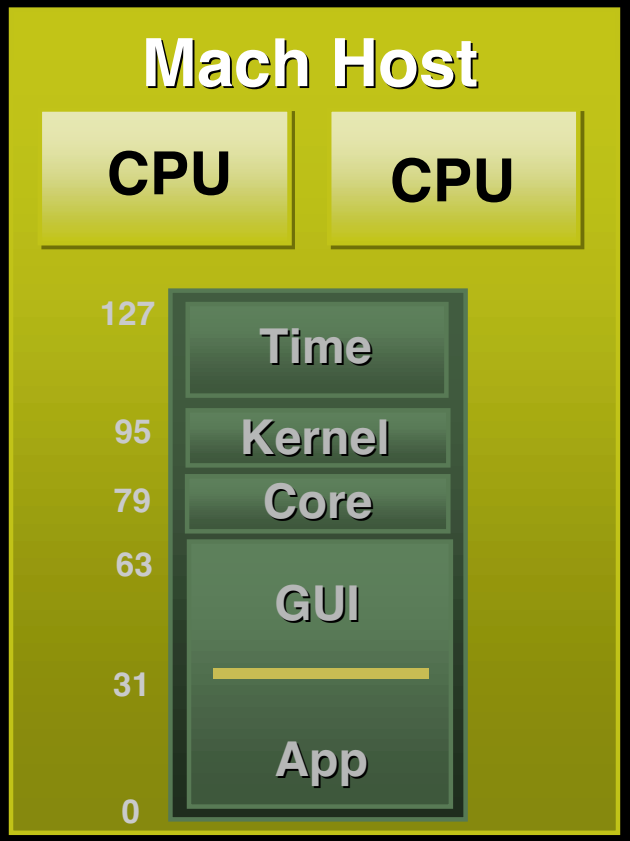
Priority bands and policies (…still subject to change)



- Time Constraint Threads
  - Developer specifies constraints
  - "watched" to assure good behavior
  - No portable interface
- GUI application threads
  - Elevated priority for responsiveness
- "Fixed" priorities privileged
  - All others adjust downward

Mach Scheduler

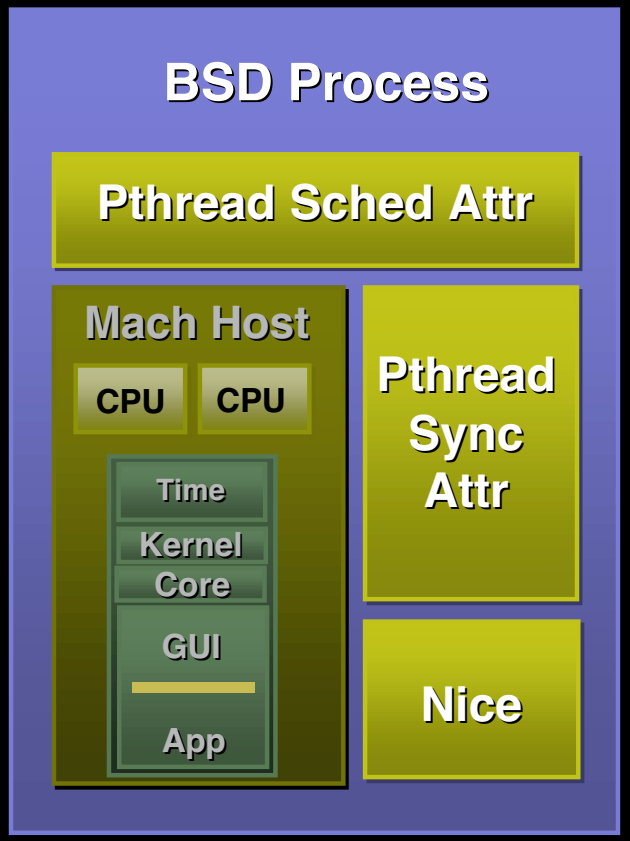
Assigns runnable threads to procesors



- Symmetric Multi-Processor (SMP)
  - Threads run on any available processor
  - Preference for affinity
- Fully Preemptive Scheduling
  - By higher priority threads when running in user mode
  - By higher priority real-time threads while running in kernel mode

BSD Scheduling Services

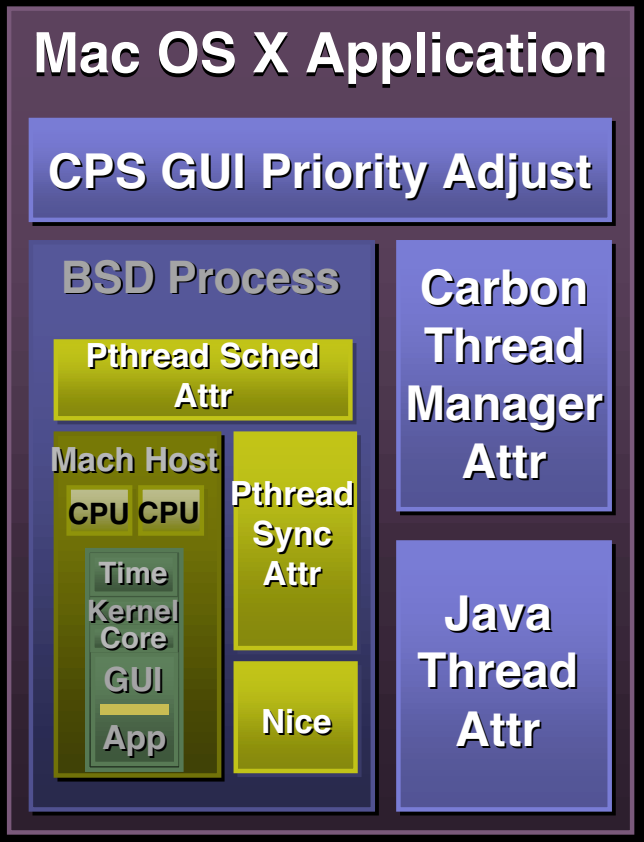
Relies heavily on Mach



- Pthread Scheduling Attributes
  - System-wide model assumes too much about the overall priority assignments
  - Only allows assignment within Application range but not outside
  - FIFO isn't safe, always treated as round-robin
- Nice value adjusts priorities
  - Task-wide

Application Scheduling Services

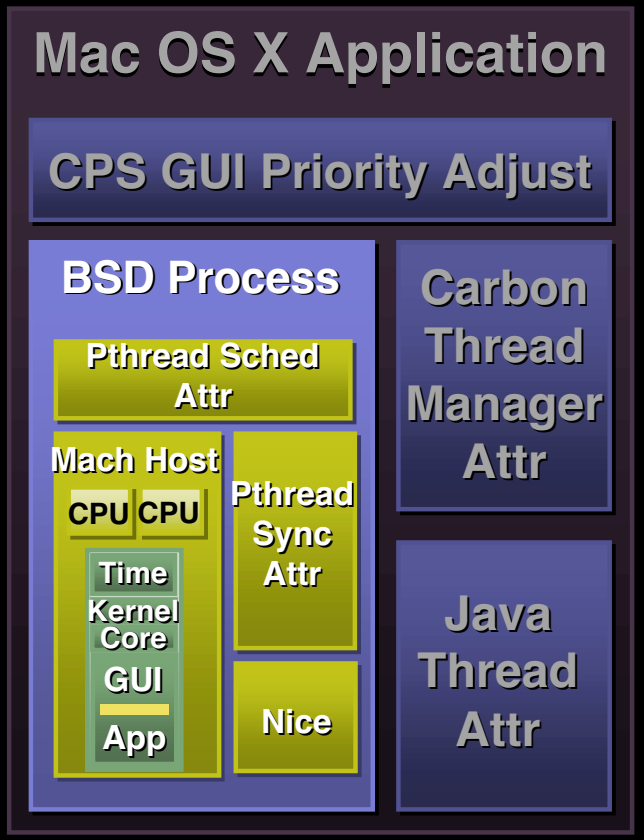
It gets quite complicated, very quickly…



- CoreServices "forward progress"
  - Assumes certain priorities are reserved to counteract stuck/spinning applications
- Carbon deferred task state, etc . . .
  - Assumes certain priorities will give traditional behavior within an application
- Each allows for time-critical threads
  - But priorities for producer/consumer threads a "black art"

Scheduling Services

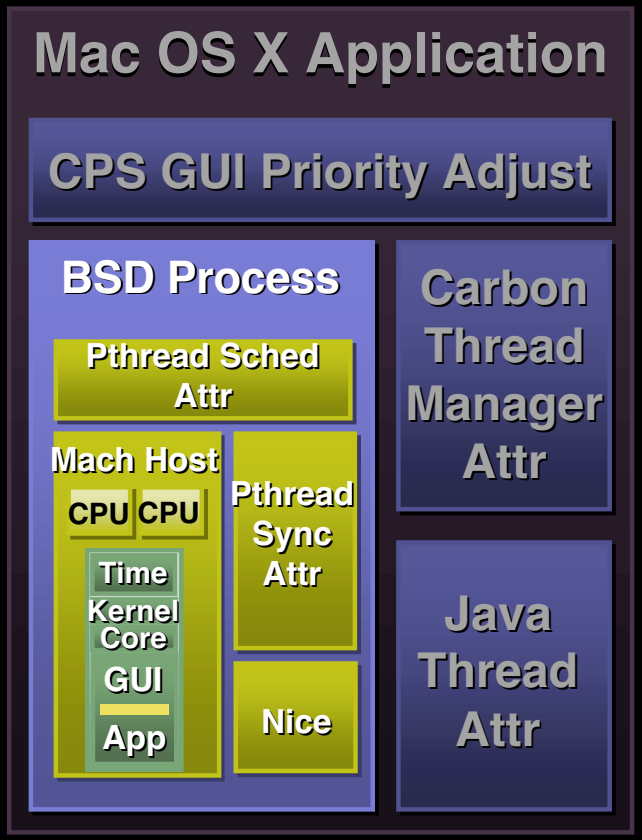
What's new from the kernel in Jaguar…



- Enhanced SMP preemption
  - Better real-time response on the second processor
- Stronger processor affinity
  - Reduced scheduler induced stress on memory system
- Rudimentary priority inheritance
  - Moves low-priority threads out of the way "quicker"
  - Kernel-only for now

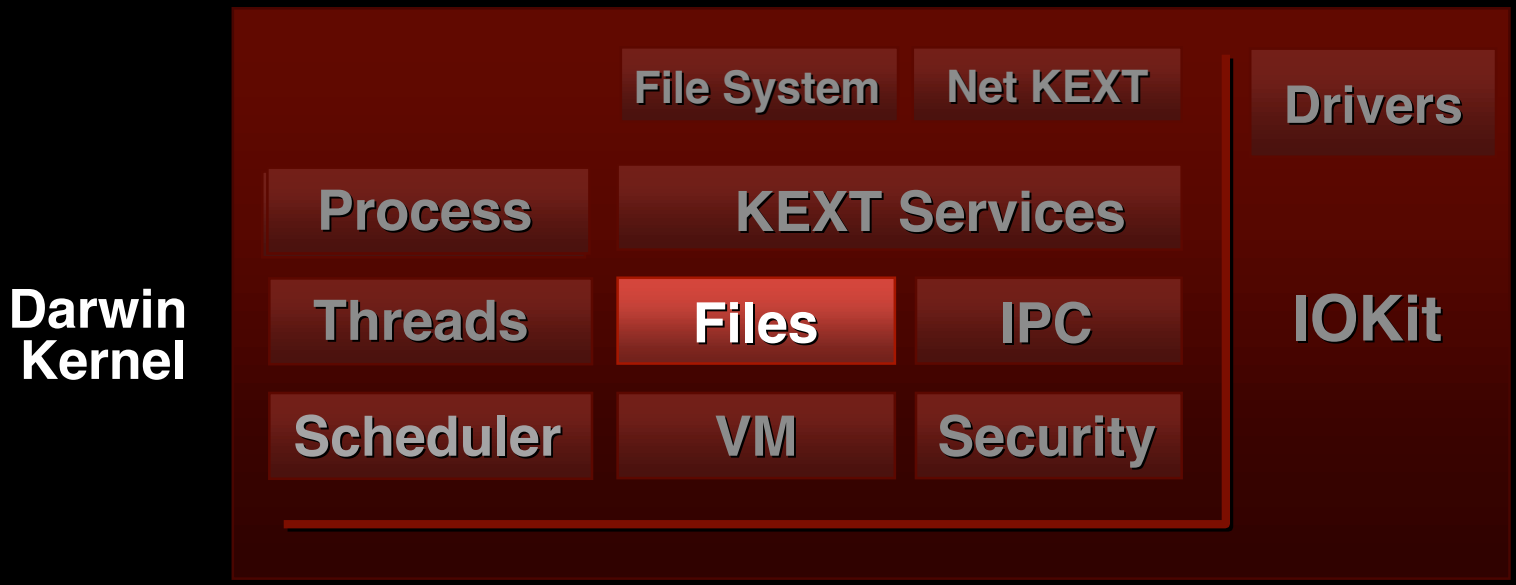
Scheduling Services

Future directions



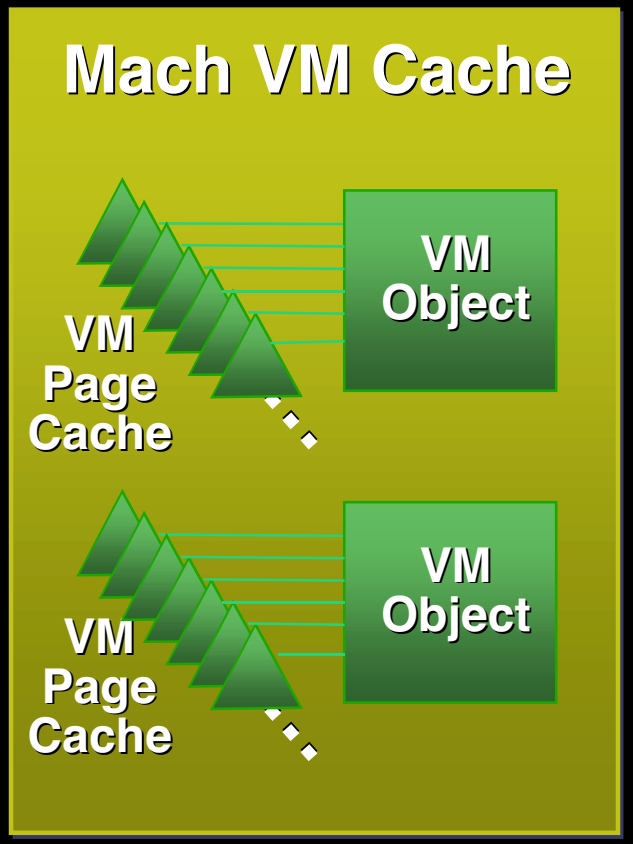
- Enhanced priority inheritance
  - Default for most synchronizers
- Wide-spread use of priority queuing
- Producer/consumer queues
  - Adjusts priorities dynamically
  - Eliminate the need for many "black art" priority assignments
- True time-critical priority calculations
  - So provide "real" values for constraints

File Services



Mach VM Object Cache

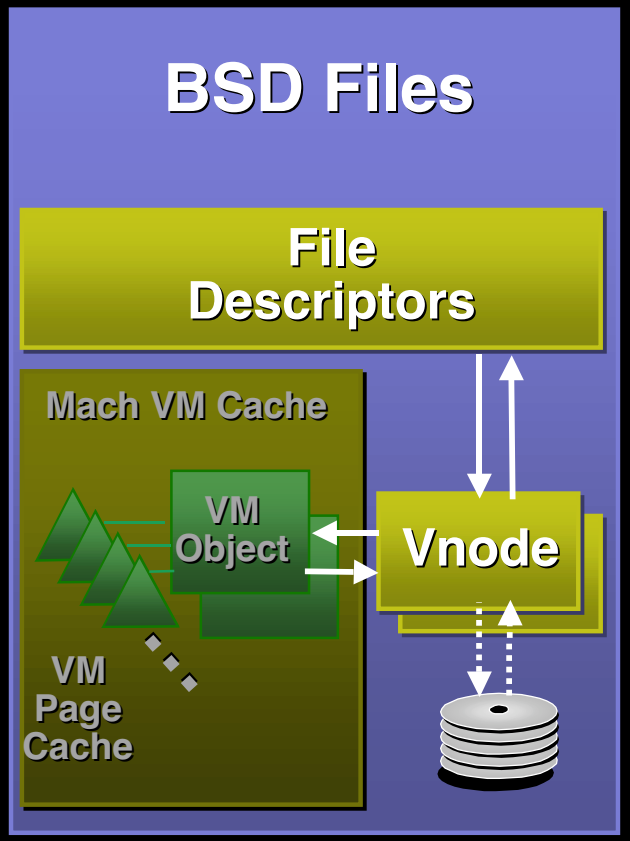
Manages all cached data



- Fills memory with cached data
  - Even though some percentage is “inactive” and not currently in use by any application
- Backing store for objects managed by pagers
  - The default pager provides backing store for “allocated” memory
- Object ranges can be mapped into task address spaces

BSD File Services

Control access to all files through descriptors



- Support traditional read-and-write semantics
  - Universal Buffer Cache (UBC) used to access/alter cached page contents
- Also supports **mmap()** semantics
  - Mapped files get direct access to those pages through Mach VM
- Clustering code used to fetch pages
  - Minimize I/O operations through pre-fetching and coalescing

Application File Services

Extend BSD/POSIX semantics

- Asynchronous operation emulation
  - BSD services are synchronous
  - A worker thread gives the appearance of async
- Arbitrate access to shared resources
  - E.g., current seek pointer per file handle
- Legacy emulation when not supported by the filesystem
  - Resource forks and catalog

File Services

What's new from the kernel for Jaguar…

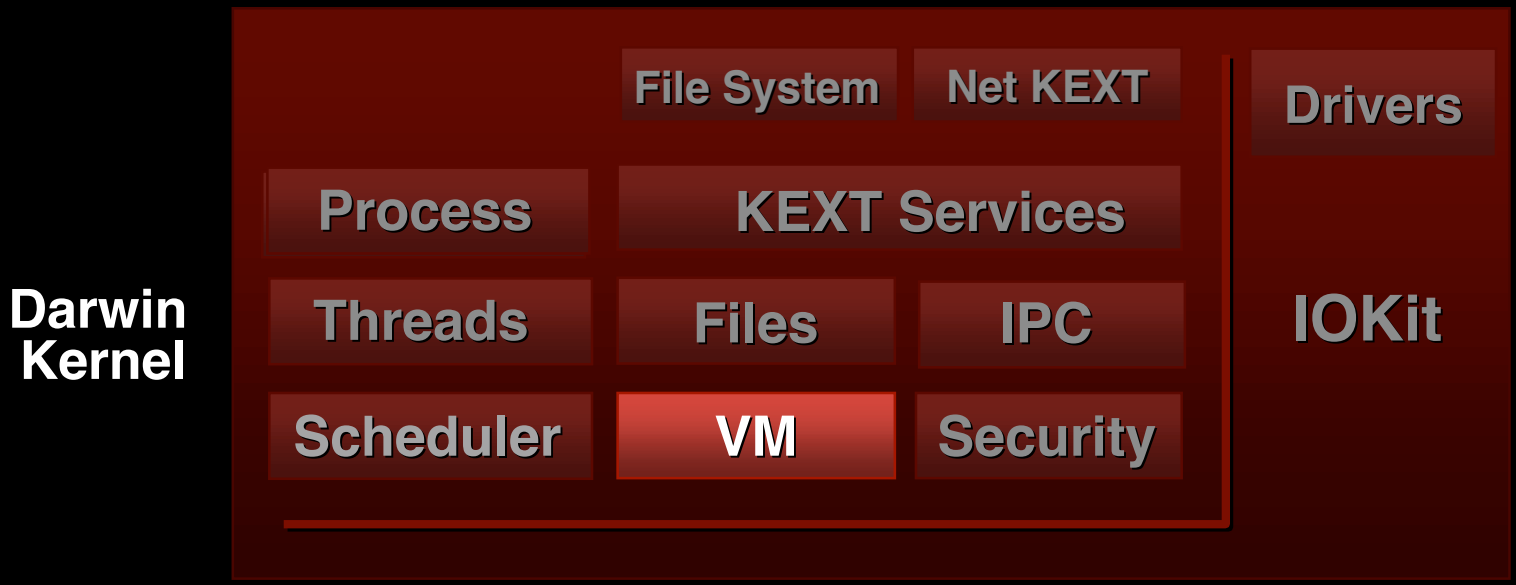
- Improved/expanded clustering
  - Dramatic reduction in I/O operations (reduced seeks)
  - Some filesystems may have to adjust as pagelists may now be even more sparse than before
- Added POSIX **pread()** and **pwrite()**
  - Avoids conflict on seek pointer

File Services

Future directions

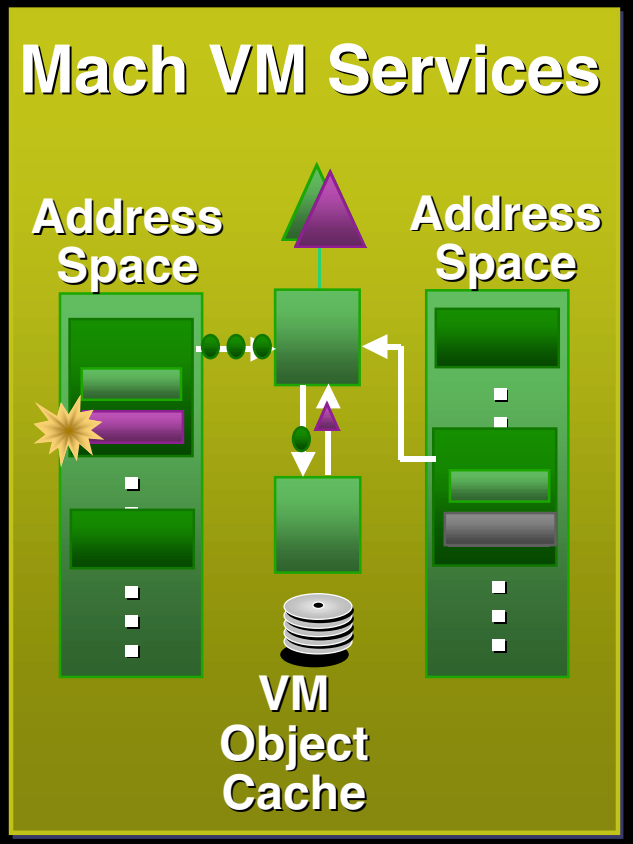
- Continued improvements in caching
  - Number of I/O operations is one of the largest contributors to a system “feeling” slow
- True async file I/O (POSIX AIO)
  - Eliminate the need for async worker threads in higher levels
- Improved concurrency in file services
  - Migrate from “funnel” to finer-grained locking for file operations

Virtual Memory Services



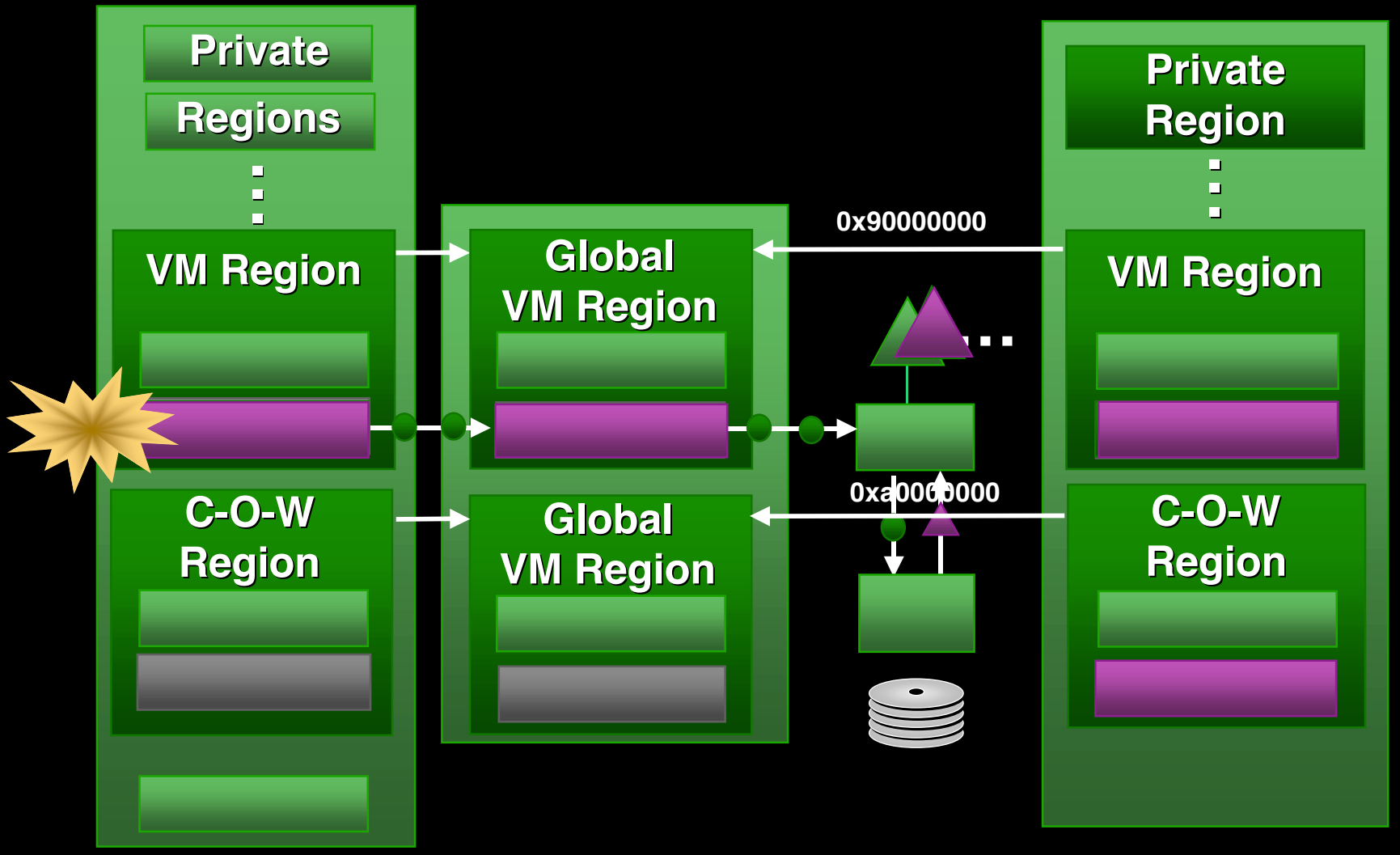
Mach Virtual Memory Services

Mach controls most aspects of virtual memory

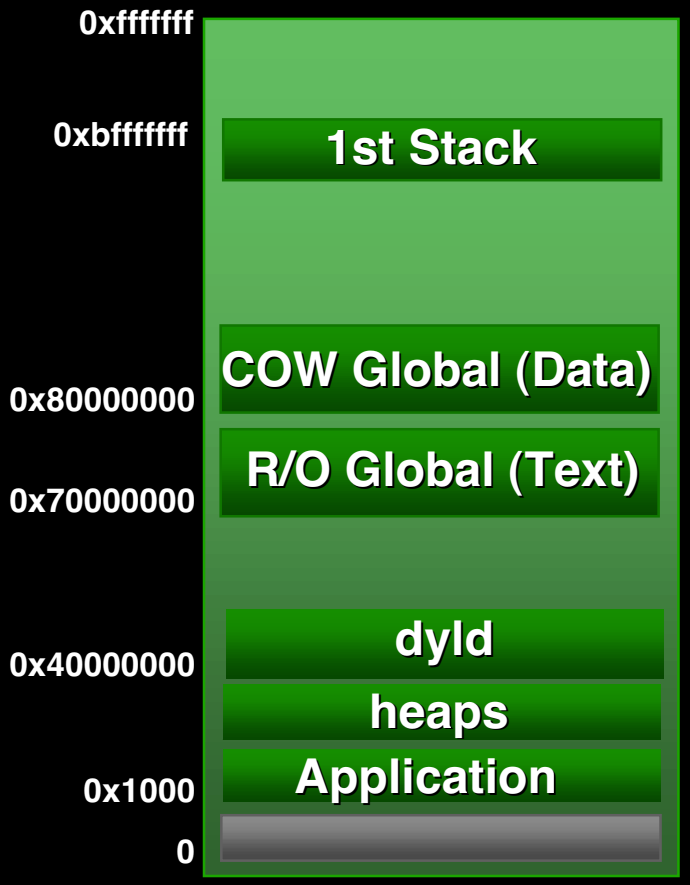


- Layout flexibility
  - Each task's address space can be constructed with a unique layout
- Protected address spaces
  - Guarded against unauthorized access or update
- Copy-on-Write optimization
- Controlled sharing
  - Single page, mapped file, to shared complex regions

Shared Region Support



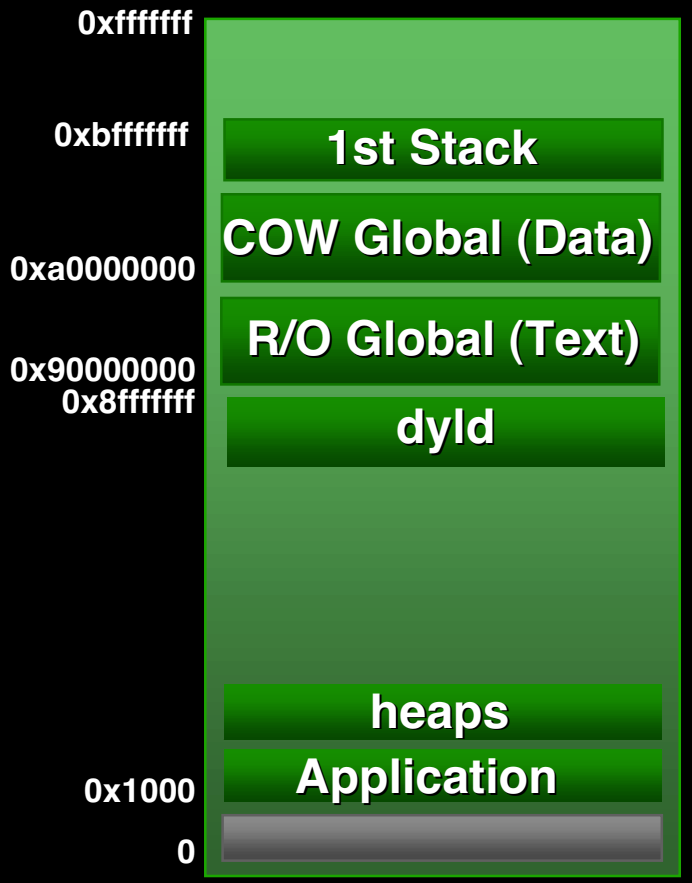
Typical Address Space Layout



- Global Shared and C-O-W Sections
  - Map system-wide frameworks and associated data into every task
  - Multiple levels of efficiency vs. mapping separately in each task
- Maximum free contiguous space was just under 1GB
  - Pretty tight for many modern applications

Virtual Memory Services

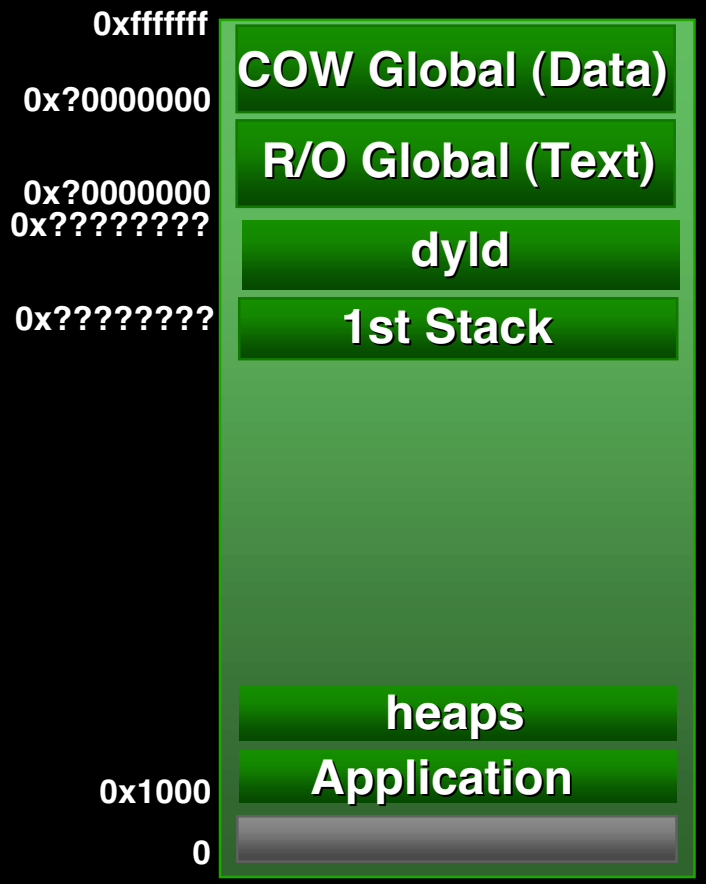
What's new in Jaguar…



- Global and dyld sections shifted
  - Expands free contiguous to just under 2GB
- Task working set optimizations
  - Pages pre-fetched based on temporal locality of reference
  - Dramatic improvement in application switching times
- Application pre-heat optimizations
  - Similar technique for App launch

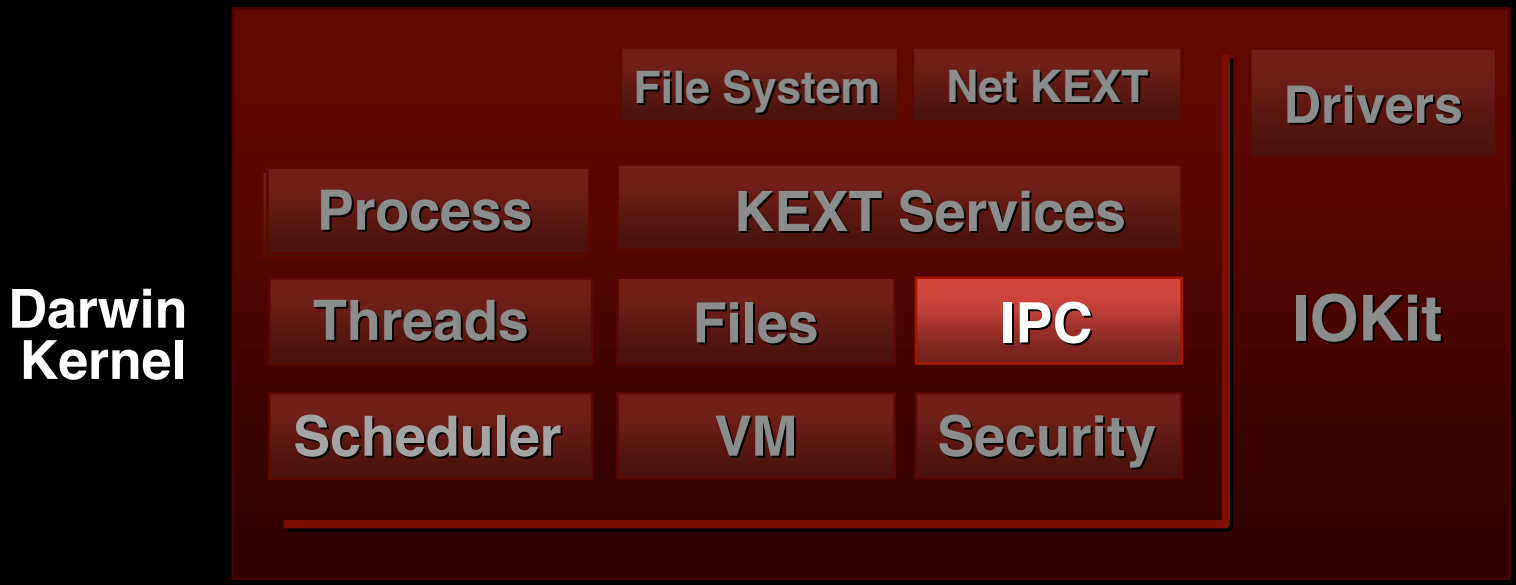
Virtual Memory Services

Future directions



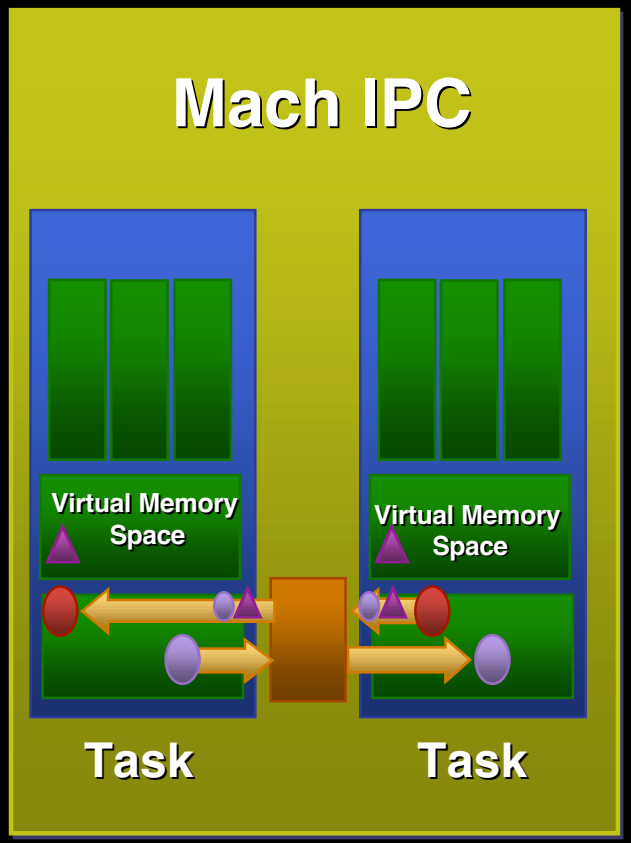
- Global and dyld sections shift again
  - Likely expand free contiguous to just under 3.5GB
- Preparation for 64-bit
  - Deprecate SPIs that make unsafe assumptions about either physical or virtual address length
- Optimizations, optimizations, etc . . .
  - Have a dramatic effect on perceived system performace

IPC Services



Mach IPC Services

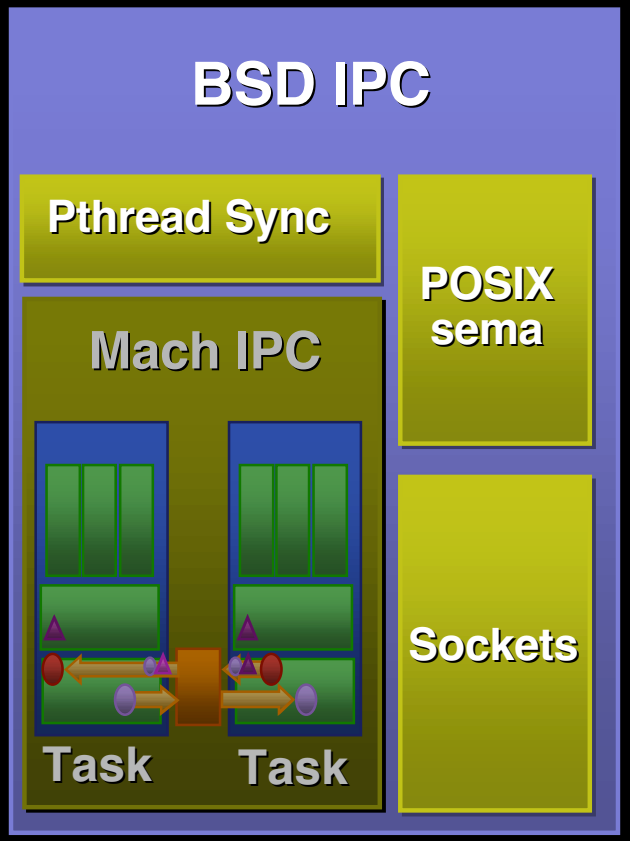
Messaging across Mach ports



- Mach ports are endpoints
  - Message queues
  - Semaphores, Locksets
- Capabilities represented by Mach port rights
  - Send, send-once
  - Receive
- Messages carry
  - Data (inline and out-of-line)
  - Additional Mach port rights
  - Sender identity information

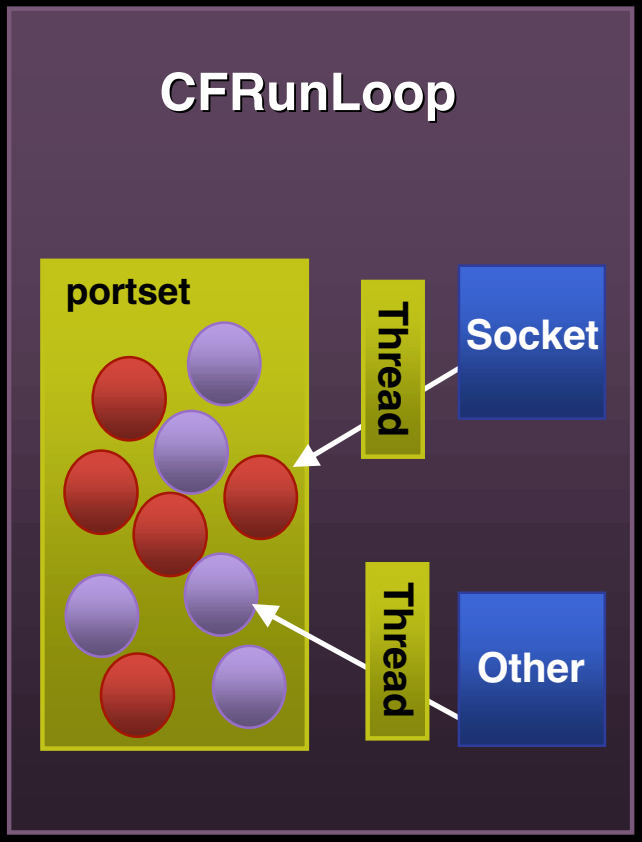
BSD IPC Services

A collection of POSIX standards



- BSD Sockets and Pipes
- POSIX semaphores
  - Named
  - Unnamed
- Pthread synchronizers
  - Hosted on Mach IPC primitives
  - So intra-process only

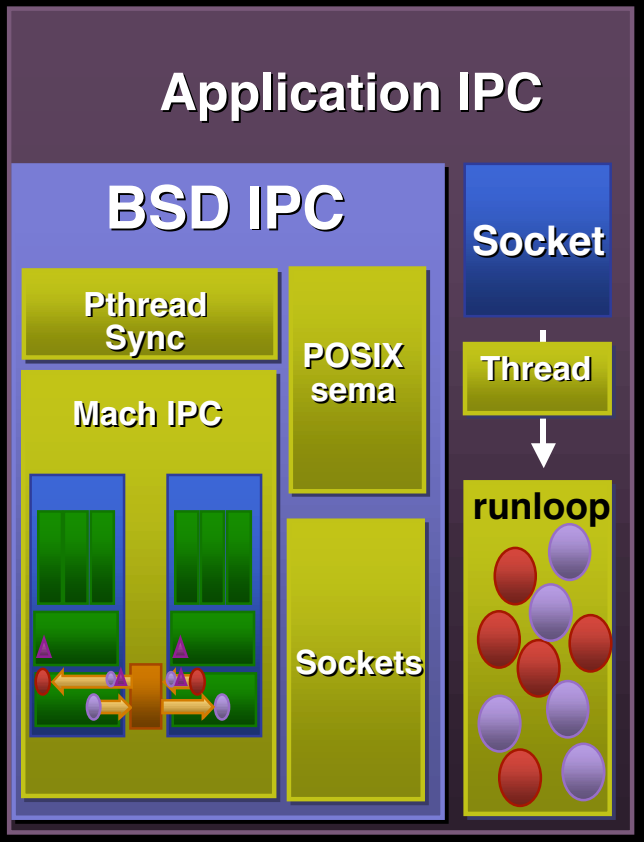
CFRunLoop Services



- Collection of port rights
  - Maintained in port sets
  - Represent event sources
- Non-port services
  - Reflected into portset from worker threads

IPC Services

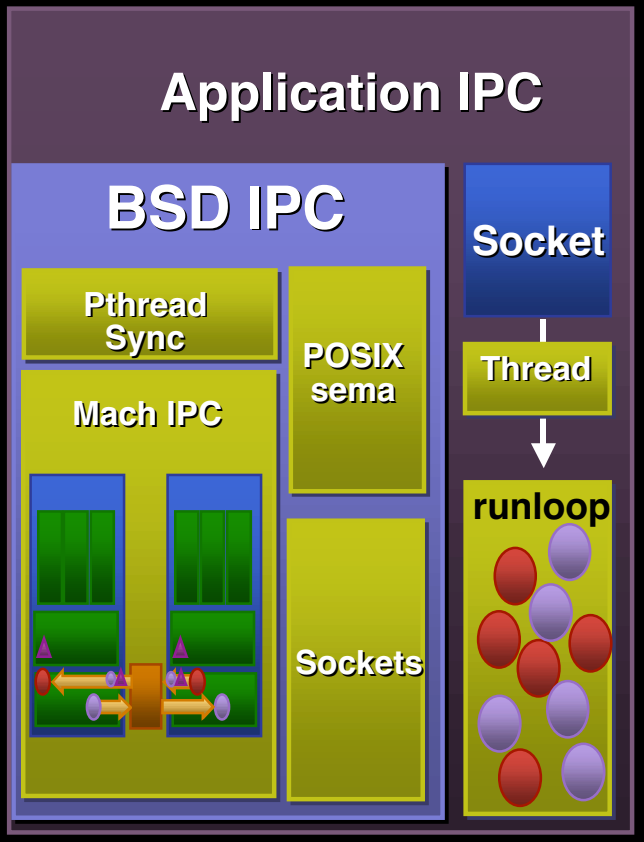
What's new in Jaguar…



- POSIX un-named Semaphores
- SystemV IPC
  - From Darwin
  - For compatibility only
  - New code should use POSIX

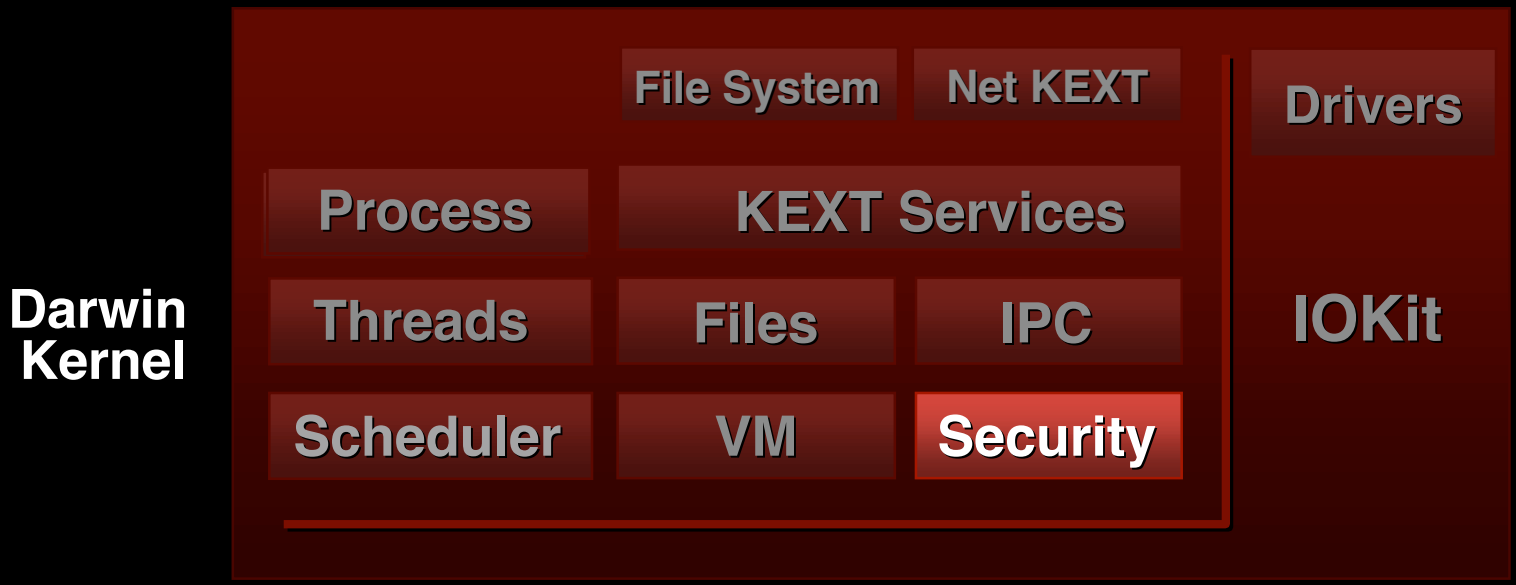
IPC Services

Future directions



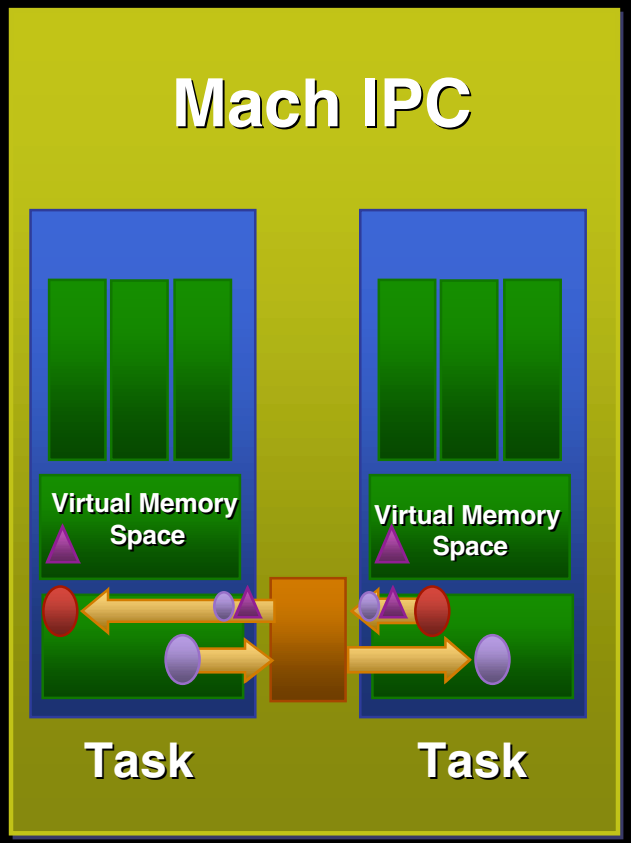
- Add **poll()** interface
- BSD kqueue support
  - Interchangable with runloops
- POSIX inter-process synchronizers
- Priority Inheritance
  - Pthread sync
  - Most others as well

Darwin Kernel Security Services



Mach Security Services

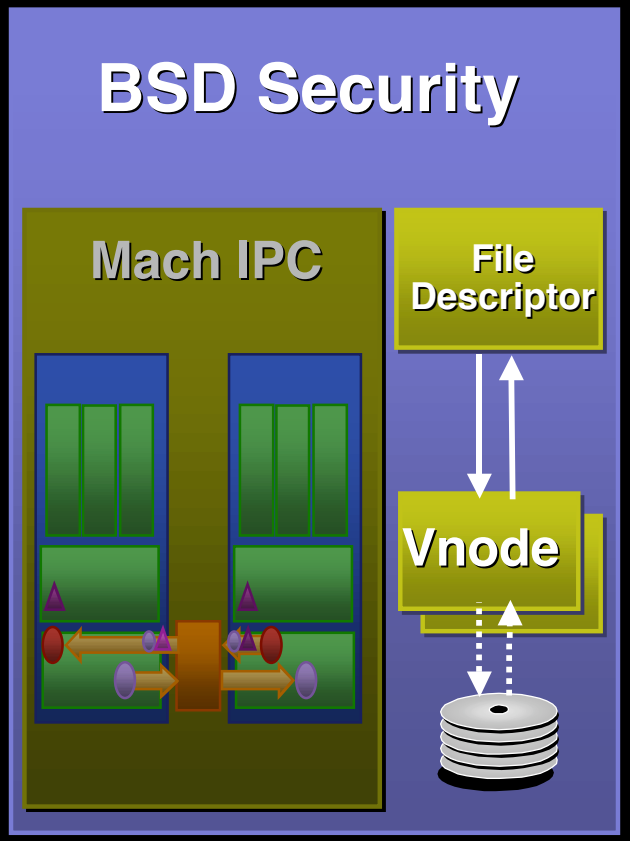
Implements mechanism—not policy



- No authentication checking
  - Strictly capabilities in the form of Mach port rights
- Divide privilege to multiple capabilities
- Sender identity tagging on each message
  - In trailer
  - Must be requested

BSD Security Services

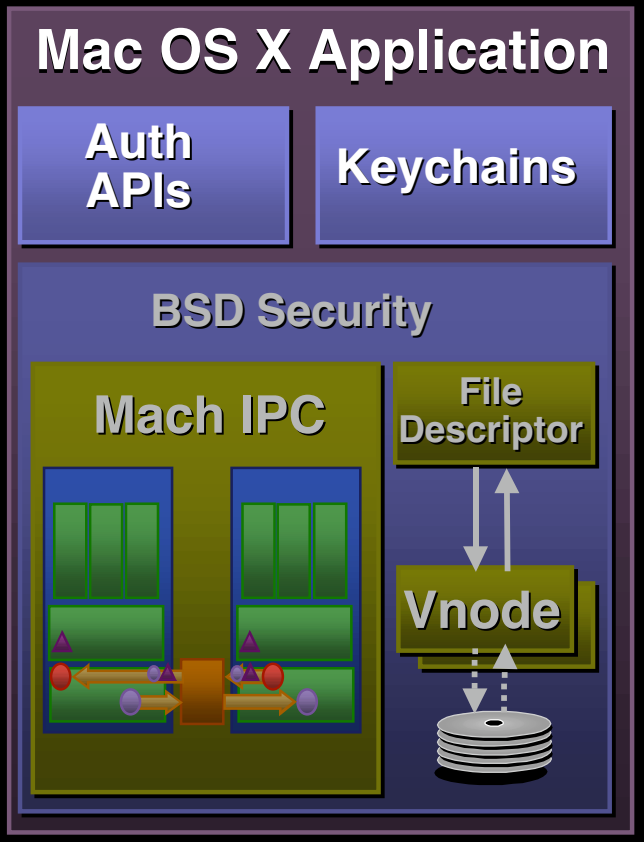
Implement most policy decisions



- User and Group IDs
  - Current credentials maintained in the kernel
- Authorization on file open
  - File descriptors cache capability
- Vnodes are data store for file permissions from filesystem

Application Security Services

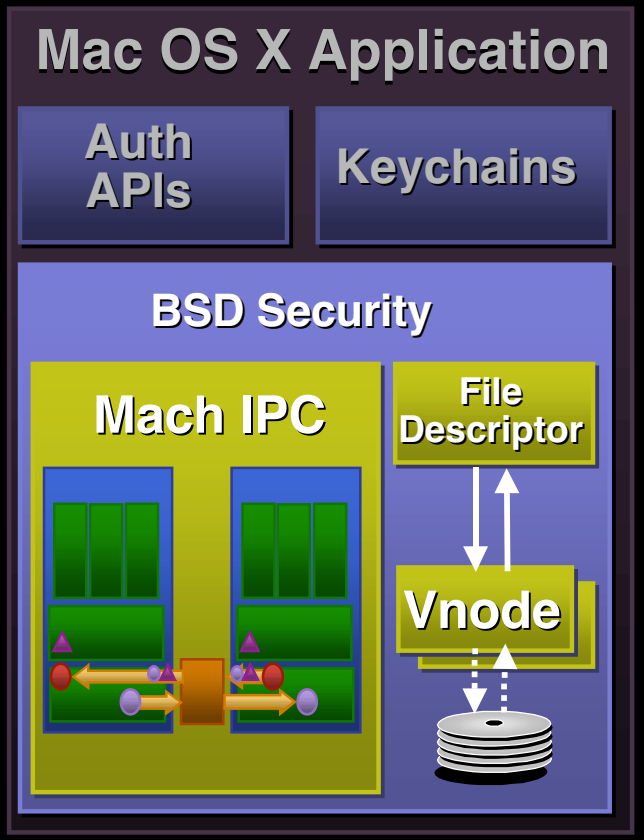
Extend BSD/POSIX semantics



- Individual Applications have unique capabilities
  - Not just users
- Keychain items
  - Maintain those cached capabilities
  - Have to be protected
- Authorization APIs
  - Privileged execution of well-defined services

Security Services

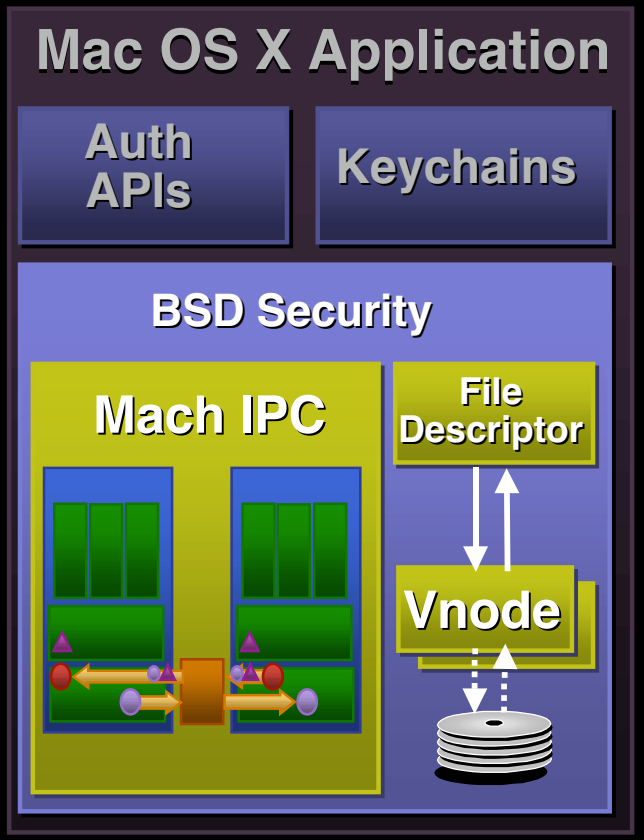
What's new from the kernel in Jaguar…



- Security Token support in the Mach Interface Generator (MIG)
  - Enhanced to support in/out security tokens on RPC calls
- CFRunLoop and others always ask for security trailers in Mach messages

Security Services

Future Directions



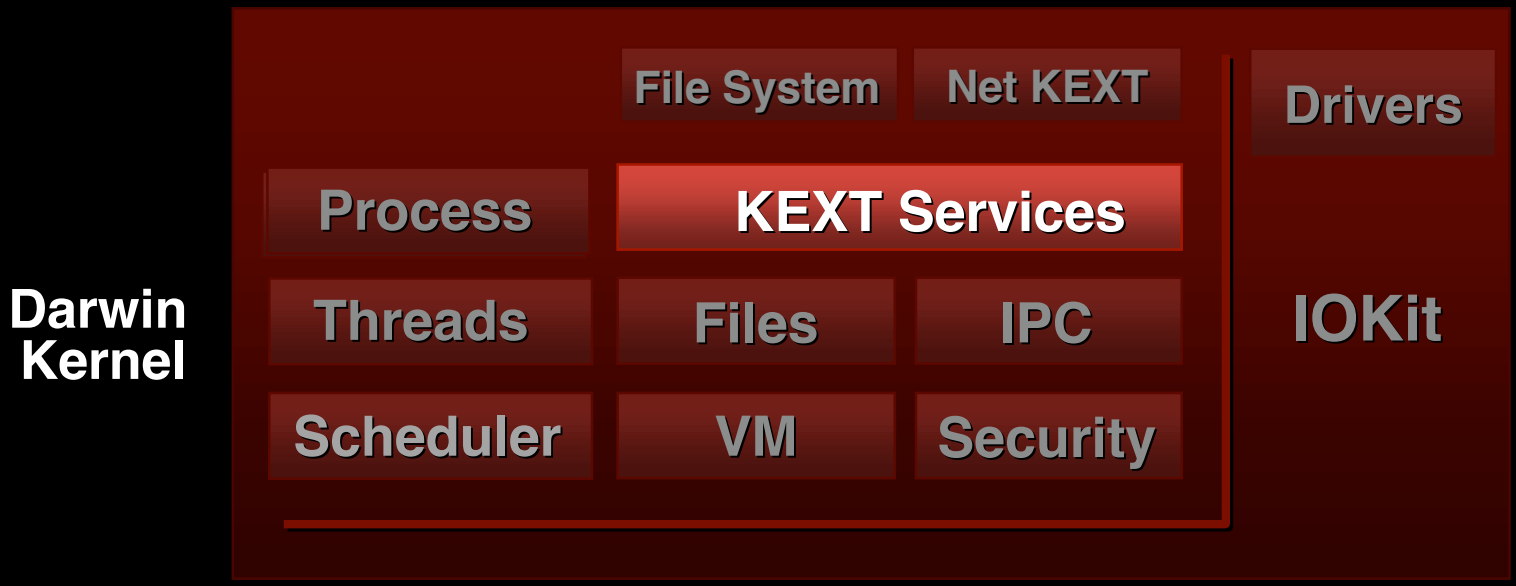
- Split Mach capabilities
  - Task and thread name ports in addition to control
- Restrict **task_for_pid()**
  - Help protect tasks from each other even under the same UID
- Migrate towards an evaluation-level of security in the kernel

Servicability Tools

What's new from the kernel in Jaguar…

- A "new" panic user experience
  - No more text scroller over the screen
  - Panic data saved off and reported by Console.app on warm restart
- Kernel debugging across routers
  - Boot-args option enables debugger to respond to ARP requests
- BSD ktrace tools

Kernel Extension Services



Darwin KEXT Services

Kernel Extensions in the past

- KEXT writing discouraged
- Most required KEXTs are IOKit Drivers
  - Kernel interfaces abstracted by families
- But what about other KEXTs?
  - Few sustainable interfaces to BSD services
  - Assumed a traditional “recompile” model
  - Unacceptable going forward

Darwin KEXT Services

What's new…

- Non-IO Kit KEXT writing highly discouraged
- Binary compatibility preserved
  - Only for a little while on non-IO Kit KEXTs
- Non-sustainable interfaces marked as private
  - Have to take overt action to compile
  - Binary interfaces likely to disappear in a future release

Darwin KEXT Services

Future directions

- Sustainable Interfaces
  - Filesystem and Network KEXTs primarily
- Defined Hooks in System Services
  - No patching of syscall table!
- Many interfaces will be location independent
  - Facilities to get the job done outside the kernel
- Send us your input!

Roadmap

108 Managing Kernel Extensions: Handling those unruly KEXTs	Civic **Wed., 10:30am**
110 Security: Authorization in Mac OS X	Civic **Wed., 2:00pm**
FF002 Darwin: Give us your input	Room J1 **Wed., 3:30pm**
112 Writing Threaded Apps: See the new pthread services in action	Room J **Thurs., 9:00am**
906 Developing for Performance: See how kernel services affect performance	Hall 2 **Fri., 9:00am**

Who to Contact

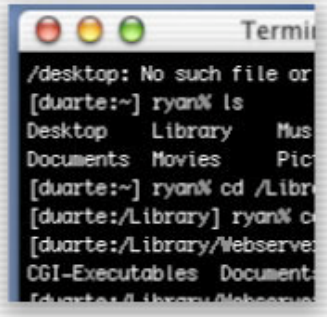
Jason Yeo
Mac OS X technology Manager
jason@apple.com

http://developer.apple.com/wwdc2002/urls.html

For More Information

- Apple Developer Website
  http://developer.apple.com
- Darwin Developer Documentation
  http://developer.apple.com/techpubs/macosx/Darwin/kernel.html
- Darwin Project Website and Mailgroups
  http://developer.apple.com/darwin/

Q&A



Jason Yeo
Mac OS X Technology Manager
jason@apple.com

http://developer.apple.com/wwdc2002/urls.html


WWDC2002


WWDC2002


WWDC2002